COMP306 Database Management Systems

Term Project - Report

Yiğithan Şanlık

Hakan Hafif

Mehmet Ulaş Uysal

İrem Arpag

7 Jun 2021

1) INTRODUCTION AND PROJECT DESCRIPTION

We created an e-commerce management application (products, employees, cities, shippers, departments, customers, suppliers, product properties, order details, categories) which has various products from many categories, orders from transformation (shippers) and suppliers to cities. Our hypothetical application is designed for the inside of the online market, so it is available also with employees and departments.

Each product on the application has a set of properties such as name, price, units in stock, buying price, supplier id … etc. For instance, SupplierID is stated in the constraint of the primary key in the Suppliers table. Additionally each employee has first name, last name, department id (also PK of the Departments table), city id (PK of Cities table in which we designed cityID and cityName), and hire date, the employee starts to work in the market. The customers and their orders can also be viewed by the manager

and the stuff. For this purpose we designed the database to be a statistical database, and the information stored will be used in relative details. There would be no significant information in our database and additionally transaction frequency and size would not lead to any problem or a performance insufficiency. The formation of our database is very scalable and it can easily be fitted to a greater number of users by slight hardware developments.

2) ENTITIES

Categories (CategoryID, CategoryName)

Shippers(ShipperID, CategoryName, CompanyName, Phone, isDeleted)

Cities(CityID, CityName)

Customers(CustomerID, FirstName, LastName, CityID, RegisterDate, isDeleted)

Departments(DepartmentID, DepartmentName)

Employees(EmployeeID, FirstName)

OrderDetails(OrderID, ProductID, UnitPrice)

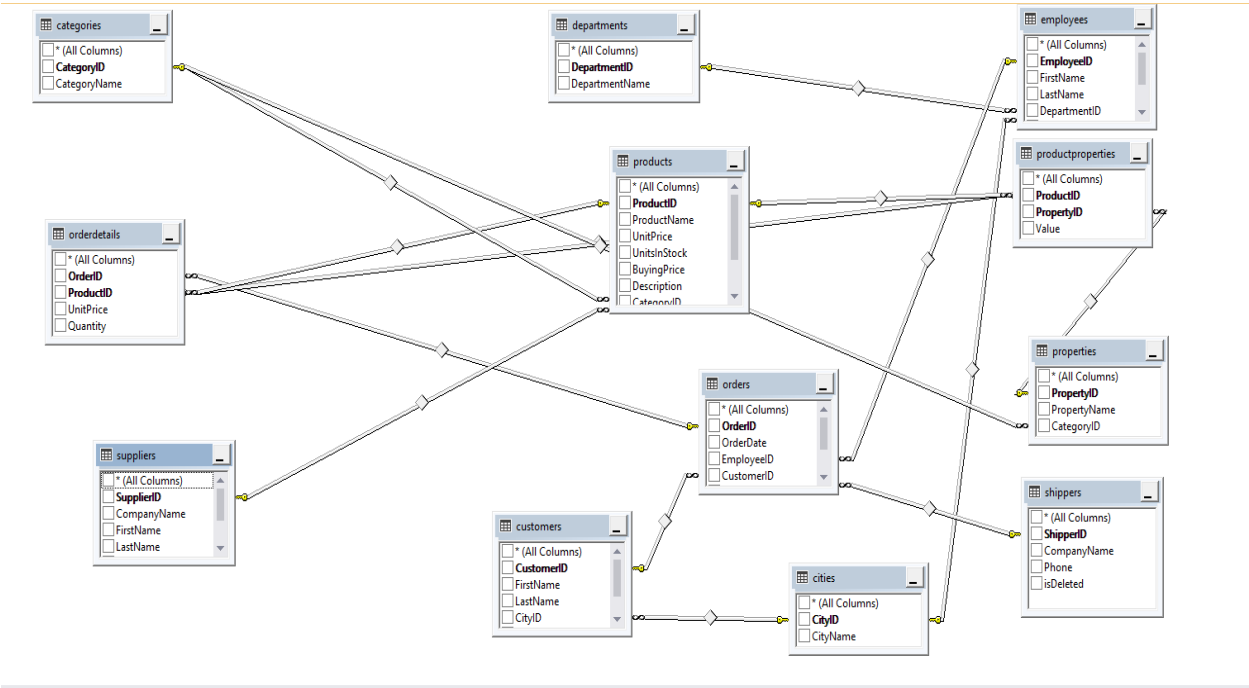
Orders(OrderID, OrderDate, EmployeeID, CustomerID, ShipperID, Freight, isCompleted)

ProductProperties(ProductID, PropertyID, Value)

Products(ProductID, ProductName, UnitPrice, UnitsInStock, BuyingPrice, Description, CategoryID, SupplierID, isDeleted)

Properties(PropertyID, PropertyName, CategoryID)

Suppliers(SupplierID, CompanyName, FirstName, LastName, Phone, Address, isDeleted)



We used Visual Studio to get that query graph. It shows us to all tables in our databases and their connections with PK - FK.

3) DATABASE CREATE TABLE STATEMENTS

CREATE TABLE Categories (CategoryID: INTEGER, CategoryName: CHAR(10), PRIMARY KEY CategoryID)

CREATE TABLE Shippers (ShipperID: INTEGER, CategoryName: CHAR(10), CityID: CHAR(15), Phone: CHAR(10), isDeleted: INTEGER, PRIMARY KEY ShipperID)

CREATE TABLE Cities (CityID: INTEGER, CityName: CHAR(10), PRIMARY KEY CityID)

CREATE TABLE Customers (CustomerID: INTEGER, FirstName: CHAR(10), LastName: CHAR(10), CityID: INTEGER, RegisterDate: CHAR(10), isDeleted: INTEGER, PRIMARY KEY CustomerID, FOREIGN KEY CityID REFERENCES Cities))

CREATE TABLE Departments (DepartmentID: INTEGER, DepartmentName: CHAR(10), PRIMARY KEY DepartmentID)

CREATE TABLE Employees (EmployeeID: INTEGER, FirstName: CHAR(10), PRIMARY KEY EmployeeID, FOREIGN KEY CityID REFERENCES Cities, FOREIGN KEY DepartmentID REFERENCES Departments)

CREATE TABLE OrderDetails (OrderID: INTEGER, ProductID: INTEGER, Quantity: INTEGER, UnitPrice: INTEGER, FOREIGN KEY OrderID REFERENCES Orders, FOREIGN KEY ProductID REFERENCES Products))

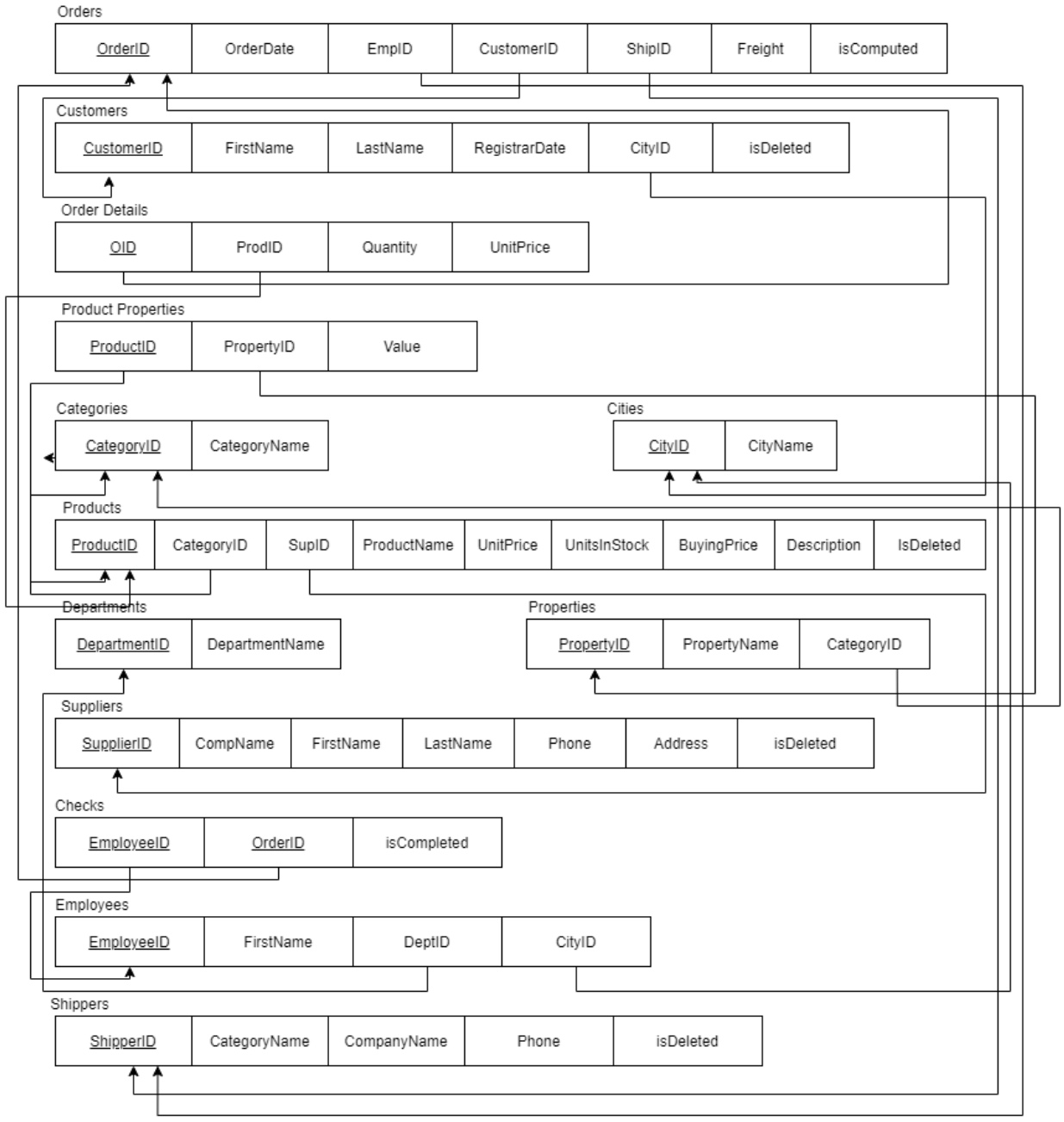
CREATE TABLE Orders (OrderID: INTEGER, OrderDate: CHAR(10), EmployeeID: INTEGER, CustomerID: INTEGER, ShipperID: INTEGER, Freight: INTEGER, isCompleted: INTEGER, PRIMARY KEY OrderID, FOREIGN KEY EmployeeID REFERENCES Employees, FOREIGN KEY CustomerID REFERENCES Customers, FOREIGN KEY ShipperID REFERENCES Shippers))

CREATE TABLE ProductProperties (ProductID: INTEGER, PropertyID: INTEGER, Value: INTEGER, FOREIGN KEY ProductID REFERENCES Products, FOREIGN KEY PropertyID REFERENCES Properties))

CREATE TABLE Products (ProductID: INTEGER,ProductName: CHAR(10), UnitPrice: INTEGER, UnitsInStock: INTEGER, BuyingPrice: INTEGER, Description: CHAR(30), CategoryID: INTEGER, SupplierID: INTEGER, isDeleted: INTEGER , PRIMARY KEY ProductID, FOREIGN KEY CategoryID REFERENCES Categories, FOREIGN KEY SupplierID REFERENCES Suppliers))

CREATE TABLE Properties (PropertyID: INTEGER,PropertyName: CHAR(10), CategoryID: INTEGER, , PRIMARY KEY PropertyID, FOREIGN KEY CategoryID REFERENCES Categories))

CREATE TABLE Suppliers (SupplierID: INTEGER,CompanyName: CHAR(10), FirstName: CHAR(10), LastName: CHAR(10), Phone: CHAR(10), Address: CHAR(30), isDeleted: INTEGER, PRIMARY KEY SupplierID)



4) DATABASE POPULATION

In essence, the database will be mostly used to see previous information, and sometimes only require particular updates for the tables. We added some entities by using insert in MySQL in the online market database. We did not use auto-generation

for populating data, since in case of randomized function may enter null values to the database even properties of the table NOT NULL point. We can as well explore the inference control designs more, and try to find out easier state-of-art approaches to implement it.

We mostly insert our datas in our own. We added some order datas from “Northwind Databases”.

5) COMPLEX QUERIES:

/* 1 */ Find the profit in which the whole products in the stock currently

```
SELECT ProductID,
    ProductName,
    UnitPrice,
    UnitsInStock,
    BuyingPrice,
    ((UnitPriceUnitsInStock)-(BuyingPriceUnitsInStock)) AS TotalProfit
FROM   Products
ORDER BY TotalProfit;
```

/* 2 */ Find the three products of the nearest of the average cost of each product in the market

```
SELECT *FROM Products
   ORDER BY  ABS(BuyingPrice - (SELECT AVG(BuyingPrice)  FROM Products)) ASC LIMIT 3;
```

/* 3 */ Find the most demanded category of the products and the total number of the products which are sold in this category

```
SELECT Categories.CategoryName,OrderDetails.Quantity count()
from Products, OrderDetails,Categories
where Products.ProductID = (OrderDetails.ProductID)and Products.CategoryID = Categories.CategoryID
group by Categories.CategoryName
order by count(*) desc;
```

/* 4 */

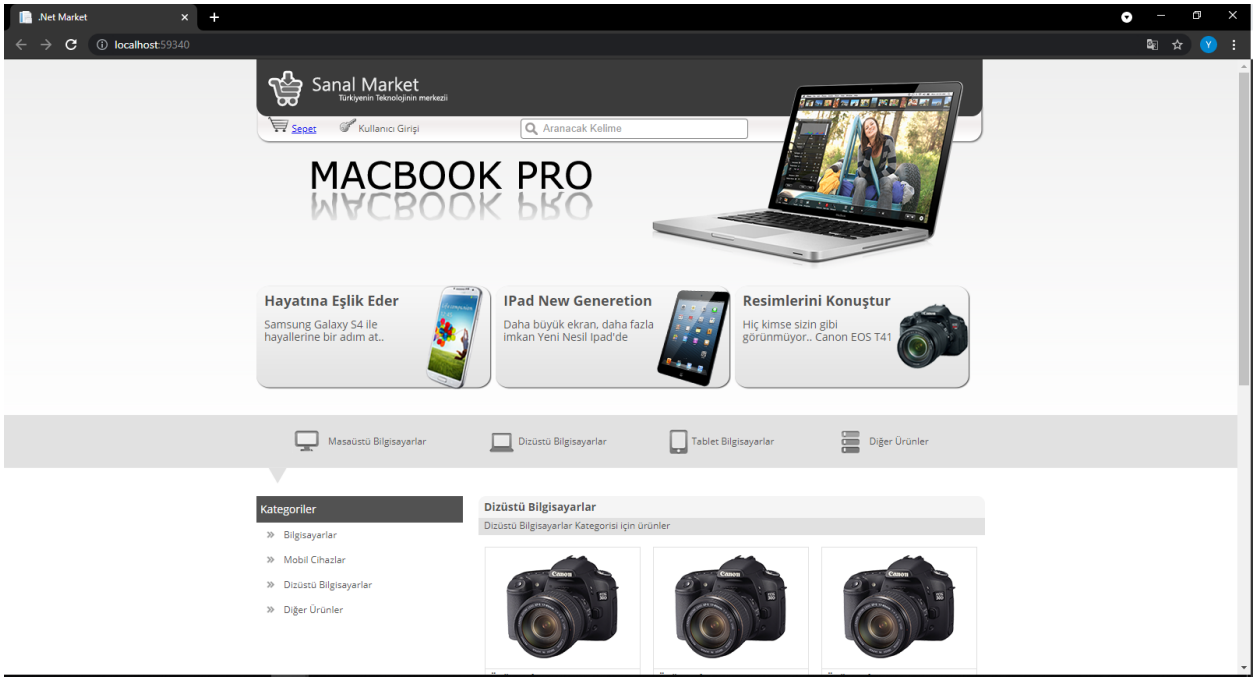
```
SELECT p.ProductName, SUM(p.UnitsInStock)
FROM Products as p
GROUP BY p.ProductName
HAVING SUM(p.UnitsInStock) >= 5000
order by SUM(p.UnitsInStock) desc;
```

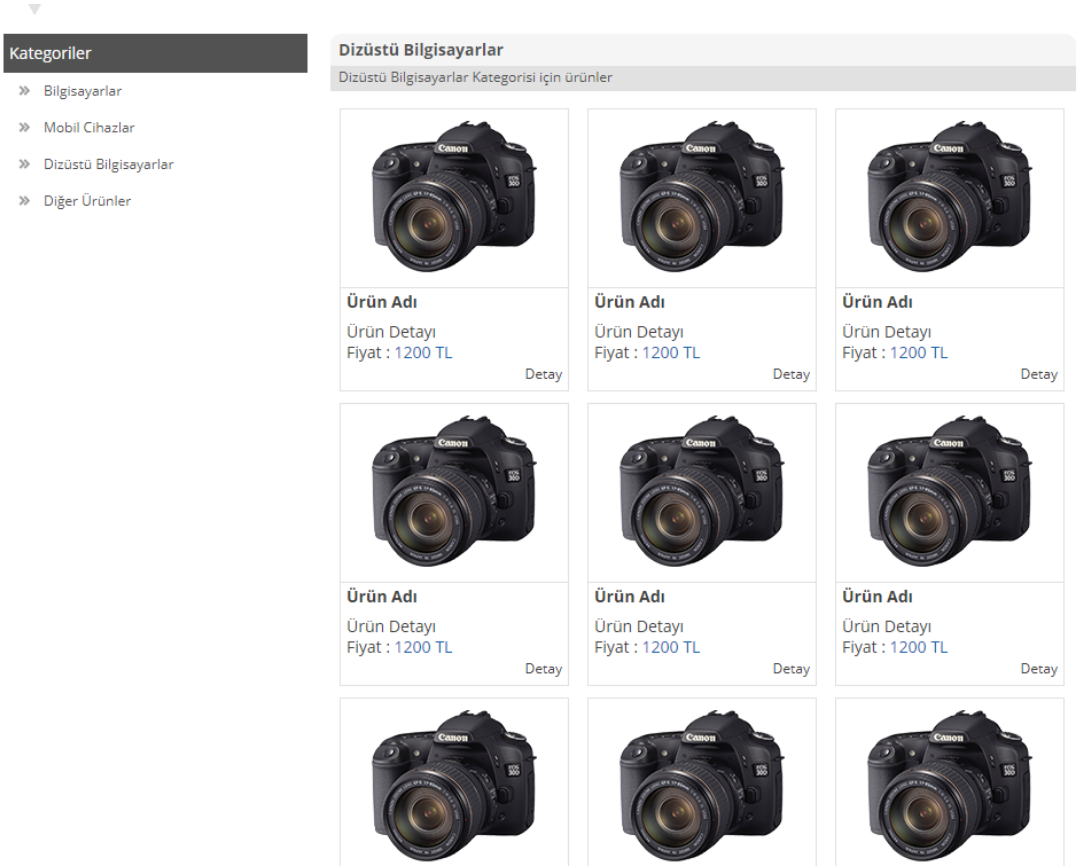
/* 5 */ Find the sum of freight of the products that are ordered by a specific customer.

```
SELECT SUM (Freight * d.Quantity)
From Orders as o, Customers as c, OrderDetails as d
Where c.CustomerID = 456798123 AND c.CustomerID = o.CustomerID AND o.OrderID = d.OrderID;
W
```

6) FINAL PROTOTYPE:

- We used mySQL databases and C# for our project. Also we used some css,javascript and html. We tried to write our code in Microsoft Visual Studio. We prepared our layout and tried to connect our database with it. We had some problems with connecting our database to our code.





Since we cannot connect our mysql database to our code. We are just able to do its layout and how our website looks like. Thus, we are only able to complete its frontend development.